Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Комп’ютерний практикум №1

Технології паралельних обчислень

Тема: Розробка потоків та дослідження пріоритету запуску потоків

Виконав

студент групи ІП-11:

Панченко С. В.

Перевірила:

Стеценко І.В.

Київ 2024

ЗМІСТ

1 ЗАВДАННЯ

1. Реалізуйте програму імітації руху більярдних кульок, в якій рух кожної кульки відтворюється в окремому потоці (див. презентацію «Створення та запуск потоків в java» та приклад). Спостерігайте роботу програми при збільшенні кількості кульок. Поясніть результати спостереження. Опишіть переваги потокової архітектури програм. 10 балів.
2. Модифікуйте програму так, щоб при потраплянні в «лузу» кульки зникали, а відповідний потік завершував свою роботу. Кількість кульок, яка потрапила в «лузу», має динамічно відображатись у текстовому полі інтерфейсу програми. 10 балів.
3. Виконайте дослідження параметру priority потоку. Для цього модифікуйте програму «Більярдна кулька» так, щоб кульки червоного кольору створювались з вищим пріоритетом потоку, в якому вони виконують рух, ніж кульки синього кольору. Спостерігайте рух червоних та синіх кульок при збільшенні загальної кількості кульок. Проведіть такий експеримент. Створіть багато кульок синього кольору (з низьким пріоритетом) і одну червоного кольору, які починають рух в одному й тому ж самому місці більярдного стола, в одному й тому ж самому напрямку та з однаковою швидкістю. Спостерігайте рух кульки з більшим пріоритетом. Повторіть експеримент кілька разів, значно збільшуючи кожного разу кількість кульок синього кольору. Зробіть висновки про вплив пріоритету потоку на його роботу в залежності від загальної кількості потоків. 20 балів.
4. Побудуйте ілюстрацію методу join() класу Thread через взаємодію потоків, що відтворюють рух більярдних кульок різного кольору. Поясніть результат, який спостерігається. 10 балів.
5. Створіть два потоки, один з яких виводить на консоль символ ‘-‘, а інший – символ ‘|’. Запустіть потоки в основній програмі так, щоб вони виводили свої символи в рядок. Виведіть на консоль 100 таких рядків. Поясніть виведений результат. 10 балів. Використовуючи найпростіші методи управління потоками, добийтесь почергового виведення на

консоль символів. 15 балів.

6. Створіть клас Counter з методами increment() та decrement(), які збільшують та зменшують значення лічильника відповідно. Створіть два потоки, один з яких збільшує 100000 разів значення лічильника, а інший – зменшує 100000 разів значення лічильника. Запустіть потоки на одночасне виконання. Спостерігайте останнє значення лічильника. Поясніть результат. 10 балів. Використовуючи синхронізований доступ, добийтесь правильної роботи лічильника при одночасній роботі з ним двох і більше потоків. Опрацюйте використання таких способів синхронізації: синхронізований метод, синхронізований блок, блокування об’єкта. Порівняйте способи синхронізації. 15 балів.

2 ВИКОНАННЯ

2.1 Більярд

Головне меню програми складається з кнопок:

- Stop — для зупинки та виходу з програми;
- Red — для створення на полі червоних кульок;
- Blue — для створення на полі синіх кульок;
- Both — для створення однієї червоної кульки та певної кількості синіх кульок;
- Random — прапор, що відповідає за поведінку створення кульок та їхньої поведінки під час відбиття від країв екрану.
- Чотирьох лунок — місця, де якщо в них потрапляє кулька, то потік її завершує свою роботу.
- Ended threads — кількість завершених потоків.

Рисунок 2.1.1 - Головне меню програми.

Наприклад під час пустого прапору Random кульки створюються у лівому краю екрана на середні та прямують у протилежну сторону, чітко відбиваюсь знову наліво.

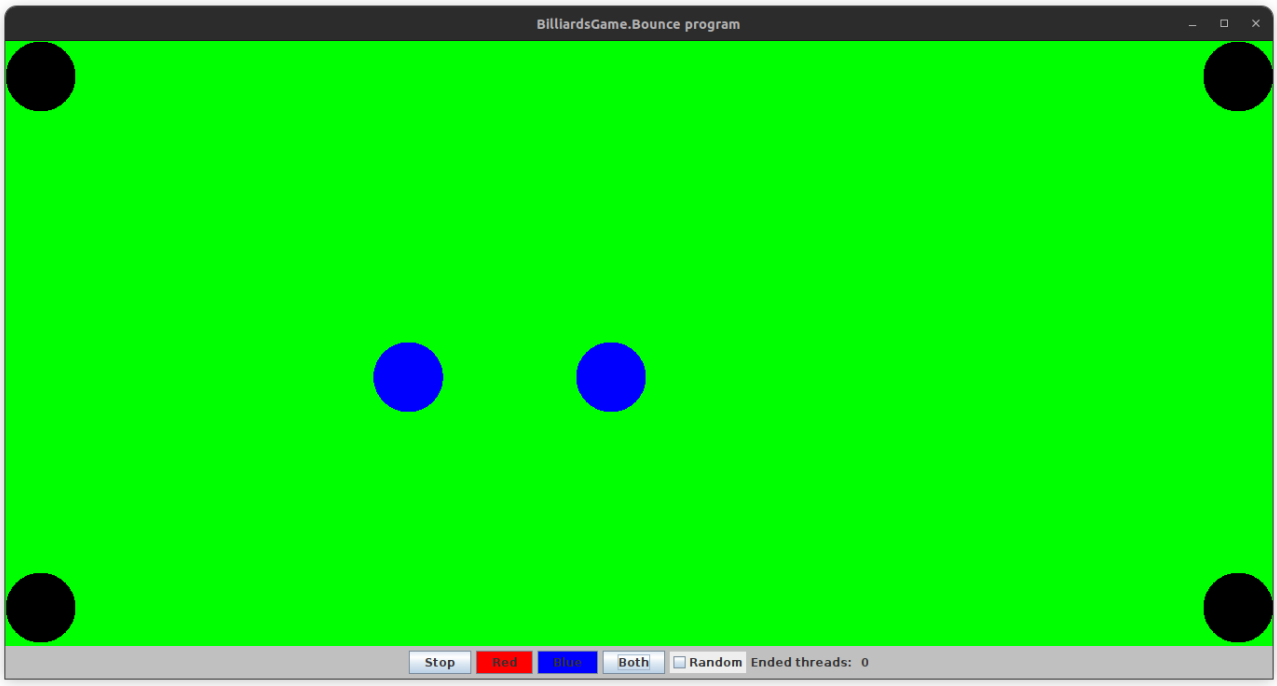

Рисунок 2.1.2 - Робота програми під час пустого прапору Random.

Продемонструємо роботу програми під час виставленого прапору Random. Побачимо, що при потраплянні в кульки поле Ended threads збільшує своє значення.

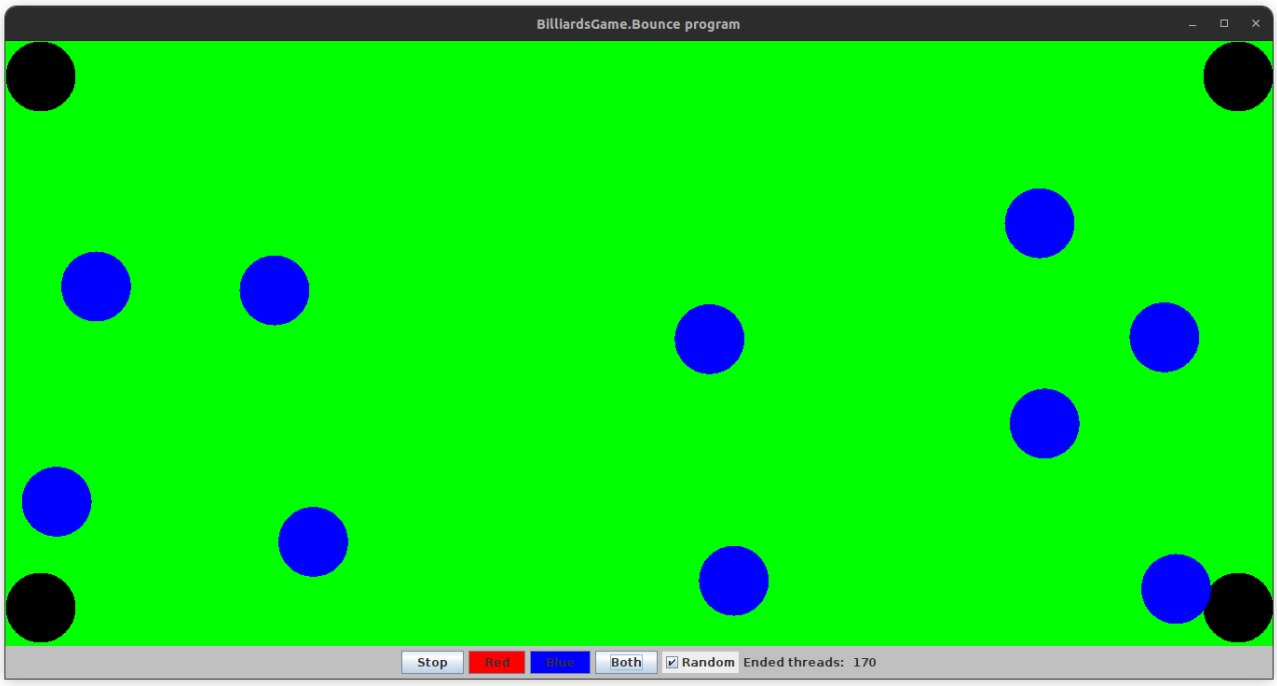

Рисунок 2.1.3 - Робота програми під час увімкненого прапору Random.

Продемонструємо пріоритетність потоку та побачимо, що попри однакову початкову швидкість та положення, червона кулька відобразилася все

таки, бо пріоритетність її потоку більша ніж відповідна їй n-кількість синіх кульок. На жаль, даний ефект можна побачити при створенні великої кількості потоків, і на даному рисунку червона кулька перекрита синіми, однак це ті сині кульки, які були створені пізніше із іншою відповідною їм червоною кулькою.

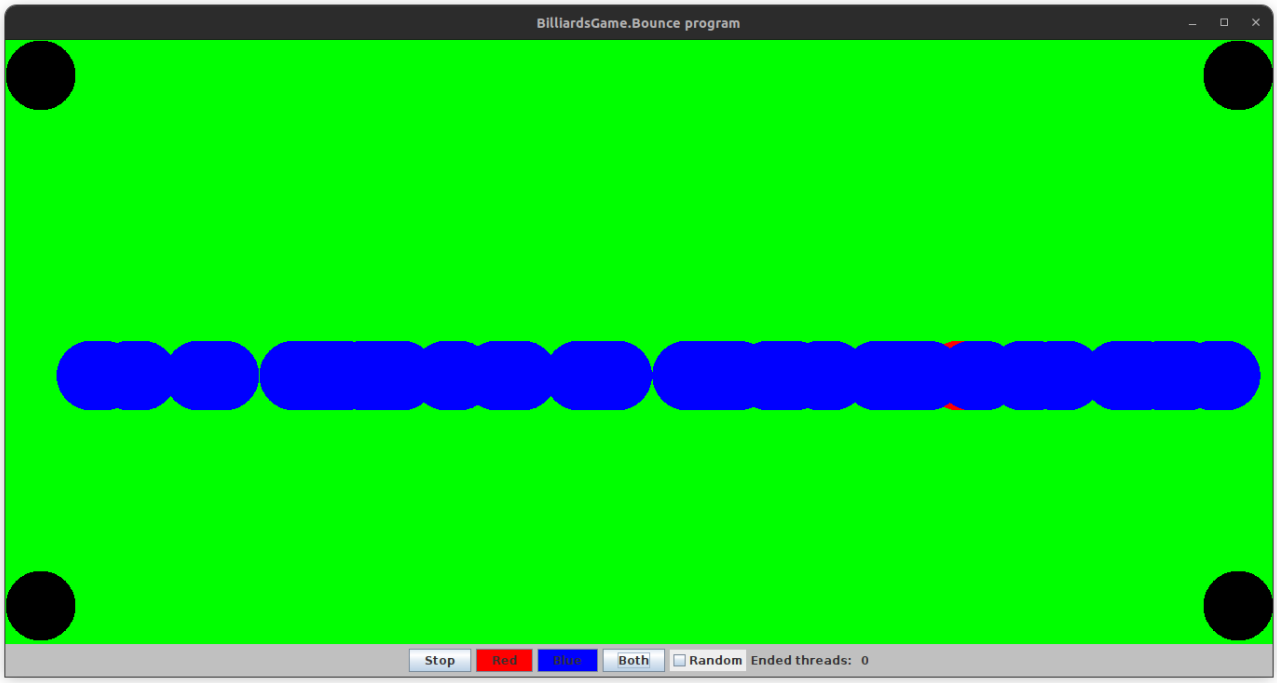

Рисунок 2.1.4 - Підтвердження роботи пріоритетності потоків.

Програма також веде логування своєї роботи. Кожен рядок містить інформацію про номер потоку; швидкість, колір, положення кульки в даний момент часу. Також є повідомлення про завершення потоку.

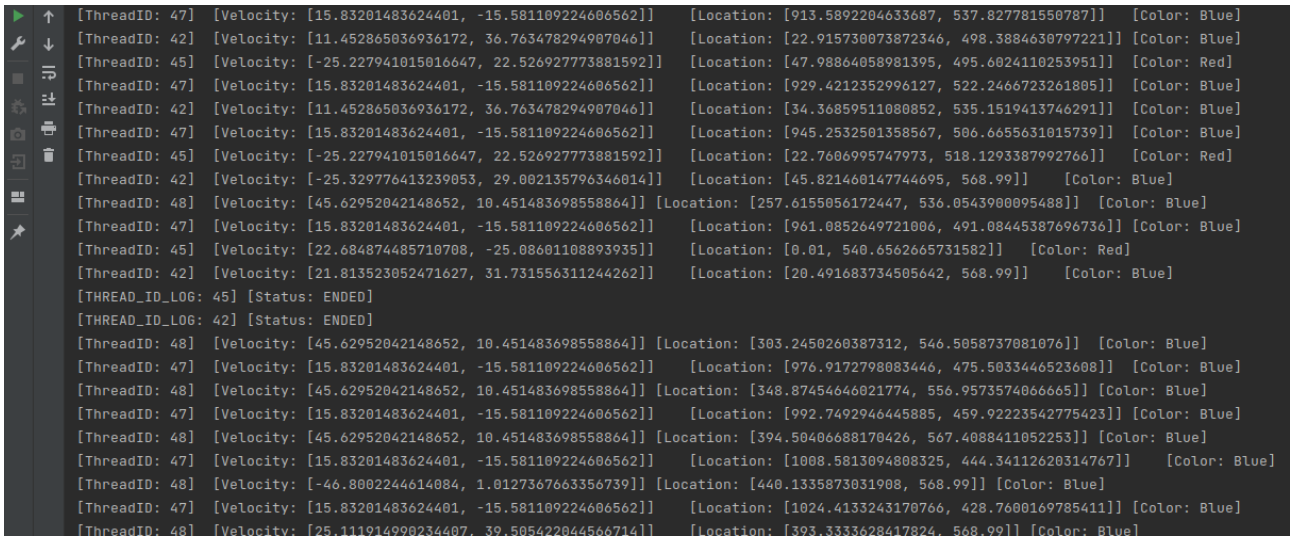

Рисунок 2.1.5 - Логування програми

Загалом структура програми реалізовує паттерн MVC(Model — View —

Control). Поглянемо на структуру.

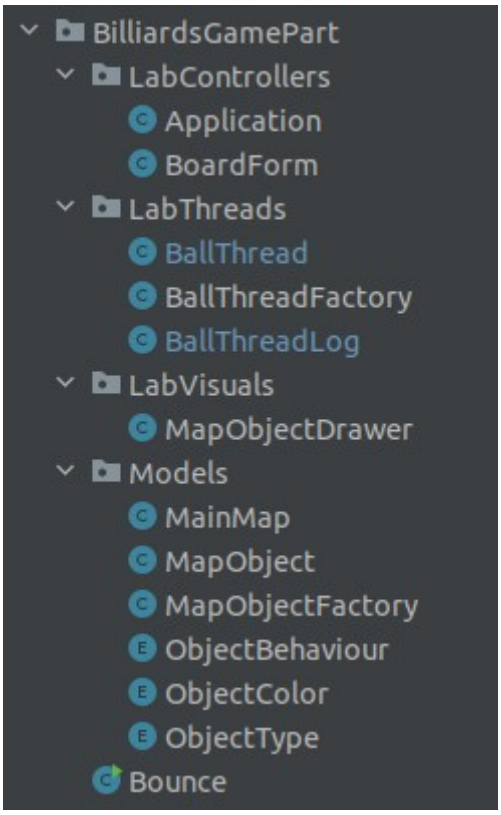

Рисунок 2.1.6 - Структура програми

Розглянемо її почергово:

1) Models — моделі об’єктів та їхні фабрики.
   - MainMap — Singleton клас, що включає в себе усі об’єкти карти.
   - MapObject — клас кульок, що перебувають на карті.
   - MapObjectFactory — фабрика кульок для контролю їхнього створення, бо загалом це залежить від поведінки кульки.
   - ObjectBehaviour — enum поведінок кульок.
   - ObjectColor — enum кольору кульок.
   - ObjectType — enum типу кульок.
2) LabVisuals — класи, що відповідають за відображення об’єктів на екран, у даному випадку кульок.
   - MapObjectDrawer — клас, що відповідає за малювання еліпсів на полотно.
3) LabControllers — форми для контролю моделей та передачі інформації на полотно.
   - BoardForm — форма, де міститься саме полотно.
   - Application — клас застосунку.

4) LabThreads — класи, що відповідають за потоки.
   - BallThread — клас, що наслідується від Thread та містить в собі посилання на відповідну кульку.
   - BallThreadFactory — фабрика, що відповідає за створення потоку, пріоритетність якого залежить від кольору кульки.
   - BallThreadLog — клас, що відповідає за логування роботи потоку.
5) Bounce — main-клас, з у якому ініціалізується об’єкт класу Application.

2.2 Виведення символів

Продемонструємо несинхронне виведення символів.

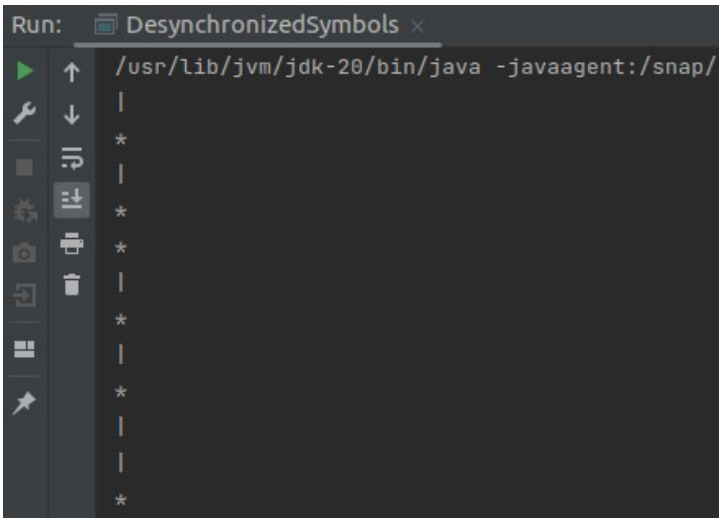

Рисунок 2.2.1 - Несинхронне виведення символів.

Продемонструємо синхронне виведення символів.

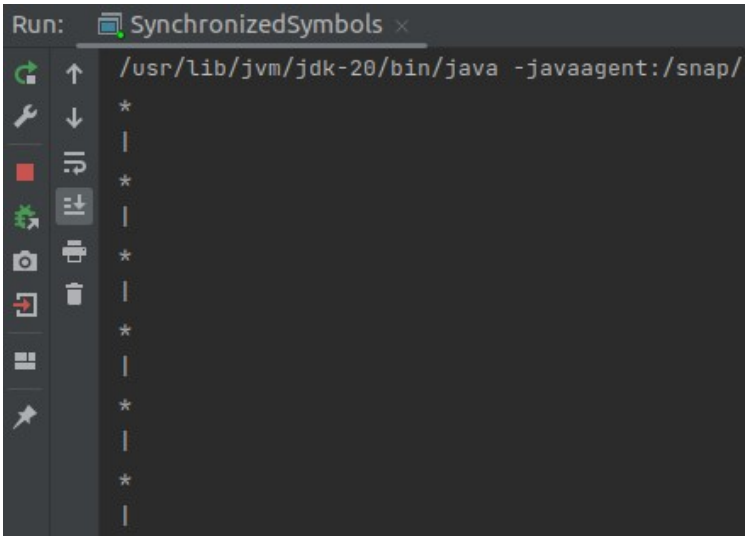

Рисунок 2.2.2 - Синхронне виведення символів.

Покажемо структуру проєкту.

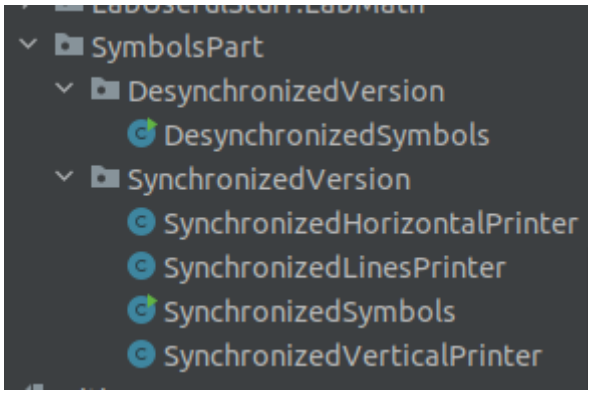

Рисунок 2.2.3 - Структура проєкту.

Розглянемо її почергово:

1) DesychronizedVersion — модуль, що містить клас, який відображає символи несинхронізовано.
   - DesychronizedSymbols — клас, який показує несинхронне зображення символів.
2) SynchronizedVersion — модуль, що відповідає за синхронне почергове відображення символів.
   1) SynchronizedLinesPrinter — клас, який має синхронізовані методи класу для відображення горизонтальної та вертикальної рисок.
   2) SynchronizedHorizontalPrinter та SynchronizedVerticalPrinter — класи, які мають поле об’єкта класу SynchronizedLinesPrinter та викликають відповідний їм метод. Ці класи виконують блокування monitor’а на даному полі, де це поле веде комунікацію між потоками через notify, забезпечуючи синхронність
   3) SynchronizedSymbols — main-клас для показу роботи програми.

2.3 Counter